# Aggregation Under Bias: Rényi Divergence Aggregation and Its Implementation via Machine Learning Markets

# Aggregation Under Bias: Rényi Divergence Aggregation and its Implementation via Machine Learning Markets

Amos J. Storkey, Zhanxing Zhu, and Jinli Hu

Institute of Adaptive Neural Computation, School of Informatics,
The University of Edinburgh, Edinburgh, EH8 9AB, UK

**Abstract.** Trading in information markets, such as machine learning markets, has been shown to be an effective approach for aggregating the beliefs of different agents. In a machine learning context, aggregation commonly uses forms of linear opinion pools, or logarithmic (log) opinion pools. It is interesting to relate information market aggregation to the machine learning setting.
In this paper we introduce a spectrum of compositional methods, Rényi divergence aggregators, that interpolate between log opinion pools and linear opinion pools. We show that these compositional methods are maximum entropy distributions for aggregating information from agents subject to individual biases, with the Rényi divergence parameter dependent on the bias. In the limit of no bias this reduces to the optimal limit of log opinion pools. We demonstrate this relationship practically on both simulated and real datasets.
We then return to information markets and show that Rényi divergence aggregators are directly implemented by machine learning markets with isoelastic utilities, and so can result from autonomous self interested decision making by individuals contributing different predictors. The risk averseness of the isoelastic utility directly relates to the Rényi divergence parameter, and hence encodes how much an agent believes (s)he may be subject to an individual bias that could affect the trading outcome: if an agent believes (s)he might be acting on significantly biased information, a more risk averse isoelastic utility is warranted.

**Keywords:** Probabilistic model aggregation, Rényi divergence, machine learning markets

## 1 Introduction

Aggregation of predictions from different agents or algorithms is becoming increasingly necessary in distributed, large scale or crowdsourced systems. Much previous focus is on aggregation of classifiers or point predictions. However, aggregation of probabilistic predictions is also of particular importance, especially where quantification of risk matters, generative models are required or where probabilistic information is critical for downstream analyses. In this paper we

focus on aggregation of probability distributions (including conditional distributions).

The problem of probabilistic aggregation in machine learning can be cast as choosing a single aggregate distribution given no (or little) direct data, but given instead the beliefs of a number of independent agents. We have no control over what these agents do, other than that we know they *do* have direct access to data and we expect them to have obtained their beliefs using that data. The data the agents observe is generated from a scenario that is the same as or similar to the target scenario we care about. We wish to choose an aggregate distribution that has high log probability under data drawn from that target scenario.

One recent approach for aggregating probabilistic machine learning predictions uses information markets [27, 28, 18] as an aggregation mechanism via the market price. In a machine learning market, agents make utility maximizing decisions regarding trades in securities. These securities are tied to the random variables of the machine learning problem. For example they could be Arrow-Debreu securities defined on each possible predicted outcome. Given the trading desires of each agent, the equilibrium price in the market then defines a distribution that is an aggregation of the beliefs of different agents. Machine learning markets combine an incentivization mechanism (to ensure agents' actions reflect their beliefs $P_i$) and a aggregation mechanism (via the trading process).

Understanding the relationship between individual actions and the aggregate market price is an interesting open question for information markets. In addition, finding efficient methods of arriving at market equilibria is key to their practical success.

The main novel contributions of this paper are

- Introducing the class of Rényi divergence based aggregators which interpolate between linear opinion pools and log opinion pools, and showing that they are the maximum entropy estimators for aggregation of beliefs potentially subject to bias. We also demonstrate this relationship practically via simulated and real problems.
- Directly relating Rényi divergence aggregators to machine learning markets with different isoelastic utilities, and showing that the risk averseness of the isoelastic utility relates to the Rényi divergence parameter that is used to control the assumed bias.

## 2 Background

Aggregation methods have been studied for some time, and have been discussed in a number of contexts. Aggregation methods differ from ensemble approaches (see e.g. [9]), as the latter also involves some control over the form of the individuals within the ensemble: with aggregation, the focus is entirely on the method of combination - there is no control over the individual agent beliefs. In addition, most aggregation methods focus on aggregating hard predictions (classifications, mean predictive values etc.) [4, 10]. Some, but not all of those

are suitable for aggregation of probabilistic predictions [7, 20], where full predictive distributions are given. This issue has received significant attention in the context of aggregating Bayesian or probabilistic beliefs [29, 8, 19, 21, 27]. Full predictive distributions are generally useful for a Bayesian analysis (where the expected loss function is computed from the posterior predictive distribution), in situations where full risk computations must be done, or simply to get the most information from the individual algorithms. Wolpert [30] describes a general framework for aggregation, where an aggregator is trained using the individual predictions on a held out validation set as inputs, and the true validation targets as outputs. This requires specification of the aggregation function. The work in this paper fits within this framework, with Rényi mixtures as the aggregator. In crowdsourcing settings, issues of reliability in different contexts come into play. Log opinion pools have been generalized to weighted log opinion pools using Bayesian approaches with an event-specific prior [17]. This emphasises that expert models can work with aggregators at many different levels, from individual data points to whole datasets within a corpus.

Recently, prediction markets, and methods derived from securities market settings [27, 28, 18, 3, 21, 7, 5], have provided a particular foundation for belief aggregation. That securities markets can perform belief aggregation was first discussed by Rubinstein [23–25]. Belief aggregation of this form is of importance in crowdsourcing settings, or settings combining information from different autonomous agents. In such settings, the beliefs of different agents can be subject to various biases.

One other area that aggregation has shown importance is in machine learning competitions, including the Netflix Challenge [14], the PASCAL Visual Object Classes challenge [11]), and many challenges set in the Kaggle challenge environment [13]. Many workshops (e.g. KDD) also run a variety of machine learning challenges. One of the most consistent take-home messages from all the challenges is that aggregation of individual entries provides a performance benefit. The final winning Netflix submission was itself a large scale aggregation of 107 different methods [22].

## 3 Problem Statement

We will postpone the discussion of information markets and start by introducing Rényi divergence aggregators and their properties, as Rényi divergence aggregators are new to this paper. We will show that Rényi divergence aggregators are intimately related to the issue of bias in individual agent beliefs.

The problem setting is as follows. We have a prediction problem to solve, in common with a number of agents. These agents have learnt probabilistic predictors on each of their own training datasets, using their own machine learning algorithms, and provide the predictions for the test scenario. We wish to combine the agents' predictions to make the best prediction we can for our setting. We don't have access to the training data the agents see, but are potentially given the held out performance of each agent on their training data, and we may have

access to their predictions for a small validation set of our own data which we know relates to our domain of interest (the distribution of which we denote by $P_G$). We consider the case where it may be possible that the data individual agents see are different in distribution (i.e. biased) with respect to our domain of interest.

Our objective is to minimize the negative log likelihood for a model $P$ for future data generated from an unknown data generating distribution $P^G$. This can be written as desiring $\arg\min_P KL(P^G||P)$, where KL denotes the KL-Divergence. However in an aggregation scenario, we do not have direct access to data that can be used to choose a model $P$ by a machine learning method. Instead we have access to beliefs $P_i$ from $i = 1, 2, \ldots, N_A$ other agents, which *do* have direct access to some data, and we must use those agent beliefs $P_i$ to form our own belief $P$.

We have no control over the agents' beliefs $P_i$, but we can expect that the agents have learnt $P_i$ using some learning algorithm with respect to data drawn from individual data distributions $P_i^G$. Hence agents will choose $P_i$ with low $KL(P_i||P_i^G)$ with respect to their individual data, drawn from $P_i^G$. For example agents can choose their own posterior distributions $P_i$ with respect to the data they observe.

We also assume that each $P_i^G$ is 'close' to the distribution $P^G$ we care about. Where we need to be specific, we use the measure $KL(P_i^G||P^G)$ as the measure of closeness, which is appropriate if $P_i^G$ is obtained by sample selection bias [26] from $P^G$. In this case $KL(P_i^G||P^G)$ gives a standardized expected log acceptance ratio, which is a measure of how the acceptance rate varies across the data distribution. Lower KL divergence means lower variation in acceptance ratio and $P_i$ is closer to $P$. The simplest case is to assume $KL(P_i^G||P^G) = 0\ \forall i$, which implies an unbiased data sample.

## 4 Weighted Divergence Aggregation

Weighted divergence-based aggregation was proposed in [12]. The idea was, given individual distributions $P_i$, to choose an aggregate distribution $P$ given by

$$P = \arg\min_Q \sum_i w_i D(P_i, Q), \tag{1}$$

where $w_i$ is a weight and $D(P_i, Q)$ represents a choice of divergence between $P_i$ and $Q$, where $D(A, B) \geq 0$, with equality iff $A = B$. This framework generalizes several popular opinion pooling methods, e.g., linear opinion pooling when $D(P_i, Q) = KL(P_i||Q)$, and log opinion pooling when $D(P_i, Q) = KL(Q||P_i)$. Concretely, a linear opinion pool is given by $P(y|\cdot) = \sum_{j=1}^{N_A} w_j P_j(y|\cdot)$, where $w_j \geq 0\ \forall j$ and $\sum_{j=1}^{N_A} w_j = 1$. The weight vector $\mathbf{w}$ can be chosen using maximum entropy arguments if we know the test performance of the individual models. Alternatively, $w_i$ can be optimized by maximizing the log likelihood of a validation set with simplex constraints, or via an expectation maximization procedure. By convexity, the solution of both optimization approaches is equivalent.

By contrast, a logarithmic opinion pool is given by $P(y|\cdot) = \frac{1}{Z(\mathbf{w})} \prod_{j=1}^{N_A} P(y|\cdot)^{w_j}$ where $w_j \geq 0 \; \forall j$, where we use the $P(y|\cdot)$ notation to reflect that this applies to both conditional and unconditional distributions. The logarithmic opinion pool is more problematic to work with due to the required computation of the normalization constant, which is linear in the number of states. Again the value of $\mathbf{w}$ can be obtained using a maximum-entropy or a gradient-based optimizer. Others (see e.g. [16]) have used various approximate schemes for log opinion pools when the state space is a product space.

Weighted Divergence aggregation is very general but we need to choose a particular form of divergence. In this paper we analyse the family of Rényi divergences for weighted divergence aggregation. This choice is motivated by two facts:

- Rényi divergence aggregators satisfy maximum entropy arguments for the aggregator class under highly relevant assumptions about the biases of individual agents.
- Rényi divergence aggregators are implemented by machine learning markets, and hence can result from autonomous self interested decision making by the individuals contributing different predictors without centralized imposition. Hence this approach can *incentivize* agents to provide their best information for aggregation.

In much of the analysis that follows we will drop the conditioning (i.e. write $P(y)$ rather than $P(y|x)$) for the sake of clarity, but without loss of generality as all results follow through in the conditional setting.

### 4.1 Weighted Rényi Divergence Aggregation

Here we introduce the family of weighted Rényi divergence methods.

**Definition 1 (Rényi Divergence).** *Let* $\mathsf{y}$ *be a random variable taking values* $y = 1, 2, \ldots, K$*. The Rényi divergence of order* $\gamma$ *(*$\gamma > 0$*) from a distribution* $P$ *to a distribution* $Q$ *is defined as*

$$D_\gamma^R[P||Q] = \frac{1}{\gamma - 1} \log \left( \sum_{y=1}^{K} P(y)^\gamma Q(y)^{1-\gamma} \right). \tag{2}$$

The Rényi divergence has two relevant special cases: $\lim_{\gamma \to 1} (1/\gamma) D_\gamma^R(P||Q) = KL(P||Q)$, and $\lim_{\gamma \to 0} (1/\gamma) D_\gamma^R(P||Q) = KL(Q||P)$ (which can be seen via L'hôpital's rule). We assume the value for the Rényi divergence for $\gamma = 1$ is defined by $KL(P||Q)$ via analytical continuation.

**Definition 2 (Weighted Rényi Divergence Aggregation).** *The* weighted Rényi divergence aggregation *is a weighted divergence aggregation given by (1), where each divergence* $D(P_i, Q) = \gamma_i^{-1} D_{\gamma_i}^R[P_i||Q]$*.*

Note that each component $i$ in (1) can have a Rényi divergence with an individualized parameter $\gamma_i$. Sometimes we will assume that all divergences are the same, and refer to a single $\gamma = \gamma_i \; \forall i$ used by all the components.

**Properties** The following propositions outline some properties of weighted Rényi divergence aggregation.

**Proposition 1.** *Weighted Rényi divergence aggregation satisfies the implicit equation for $P(y)$ of*

$$P(y) = \frac{1}{Z} \sum_i w_i \gamma_i^{-1} \frac{P_i(y)^{\gamma_i} P(y)^{1-\gamma_i}}{\sum_{y'} P_i(y')^{\gamma_i} P(y')^{1-\gamma_i}} \tag{3}$$

*where $w_i$ are given non-negative weights, and $Z = Z(\{\gamma_i\}) = \sum_i w_i \gamma_i^{-1}$ is a normalisation constant, and $\{\gamma_i\}$ is the set of Rényi divergence parameters.*

*Proof.* Outline: Use $D(P_i, Q) = \gamma_i^{-1} D^R_{\gamma_i}[P_i||Q]$ from (2) in Equation (1), and build the Lagrangian incorporating the constraint $\sum_y Q(y) = 1$ with Lagrange multiplier $Z$. Use calculus of variations w.r.t. $Q(y)$ to get $K$ equations

$$\sum_i w_i \gamma_i^{-1} \frac{P_i(y)^{\gamma_i} P(y)^{-\gamma_i}}{\sum_{y'=1}^K P_i(y')^{\gamma_i} P(y')^{1-\gamma_i}} - Z = 0 \tag{4}$$

for the optimum values of $P(y)$. Multiply each equation with $P(y)$ and find $Z = \sum_j w_j \gamma_j^{-1}$ by summing over all equations. Rearrange to obtain the result.

**Proposition 2.** *Weighted Rényi divergence aggregation interpolates between linear opinion pooling ($\gamma \to 1$) and log opinion pooling ($\gamma \to 0$).*

*Proof.* Outline: Set $\gamma_i = 1$ in (3) to obtain a standard linear opinion pool. For log opinion pool, set $\gamma_i = \gamma$, and take $\gamma \to 0$. Note (3) can be written $Z = \sum_i w_i \gamma_i^{-1} \frac{\partial}{\partial Q} D^R_{\gamma_i}[P_i||Q]$. Using L'Hôpital's rule on each element in the sum and switching the order of differentiation $(\partial/\partial\gamma_i)(\partial/\partial Q) = (\partial/\partial Q)(\partial/\partial\gamma_i)$ gives the result.

In the next section we show that Rényi divergence aggregation provides the maximum entropy distribution for combining together agent distributions where the belief of each agent is subject to a particular form of bias. Two consequences that are worth alerting the reader to ahead of that analysis are:

1. If all agents form beliefs on data drawn from the same (unbiased) distribution then the maximum entropy distribution is of the form of a log opinion pool.
2. If all agents form beliefs on unrelated data then the maximum entropy distribution is of the form of a linear opinion pool.

## 5 Maximum Entropy Arguments

Consider the problem of choosing an aggregator distribution $P$ to model an unknown target distribution $P^G$ given a number of individual distributions $P_i$. These individual distributions are assumed to be learnt from data by a number of individual agents. We will assume the individual agents did not (necessarily)

have access to data drawn from $P^G$, but instead the data seen by the individual agents was biased, and instead sampled from distribution $P_i^G$. In aggregating the agent beliefs, we neither know the target distribution $P^G$, nor any of the individual bias distributions $P_i^G$, but model them with $P$ and $Q_i$ respectively.

As far as the individual agents are concerned they train and evaluate their methods on their individual data, unconcerned that their domains were biased with respect to the domain we care about. We can think of this scenario as convergent dataset shift [26], where there is a shift from the individual training to a common test scenario. The result is that we are given information regarding the test log likelihood performance for each $P_i$ in their own domains: $\sum_y P^G(y) \log P_i(y) = a_i$.

The individual agent data is biased, not unrelated, and so we make the assumption that the individual distributions $P_i^G$ are related to $P$ in some way. We assume that $KL(P_i^G||P^G)$ is subject to some bound (and call this the nearness constraint). As mentioned in the Problem Statement this is a constraint on the standardized expected log acceptance ratio, under an assumption that $P_i^G$ is derived from $P^G$ via a sample selection bias.

Given this scenario, a reasonable ambition is to find maximum entropy distributions $Q_i$ to model $P_i^G$ that capture the performance of the individual distributions $P_i$, while at the same time being related via an unknown distribution $P$. As we know the test performance, we write this as the constraints:

$$\sum_y Q_i(y) \log P_i(y) = a_i, \tag{5}$$

The nearness constraints[1] for $Q_i$ are written as

$$KL(Q_i||P) \leq A_i \tag{6}$$

$$\Rightarrow \sum_y Q_i(y) \log \frac{Q_i(y)}{P(y)} \leq A_i \text{ for some } P\ . \tag{7}$$

encoding that our model $Q_i$ for $P_i^G$ must be near to the model $P$ for $P^G$. That is the KL divergence between the two distributions must be bounded by some value $A_i$.

Given these constraints, the maximum entropy (minimum negative entropy) Lagrangian optimisation can be written as $\arg\min_{\{Q_i\},P} L(\{Q_i\}, P)$, where

$$\begin{aligned} L(\{Q_i\}, P) = &\sum_i \sum_y Q_i(y) \log Q_i(y) + \sum_i b_i (1 - \sum_y Q_i(y)) \\ &- \sum_i \lambda_i \left( \left[ \sum_y Q_i(y) \log P_i(y) \right] - a_i \right) + c(1 - \sum_y P(y)) \\ &+ \sum_i \rho_i \left( \left[ \sum_y Q_i(y) \log \frac{Q_i(y)}{P(y)} \right] - A_i + s_i \right) \end{aligned} \tag{8}$$

[1] We could work with a nearness penalty of the same form rather than a nearness constraint. The resulting maximum entropy solution would be of the same form.

where $s_i$ are slack variables $s_i \geq 0$, and $\rho_i, \lambda_i, b_i$ and $c$ are Lagrange multipliers. This minimisation chooses maximum entropy $Q_i$, while ensuring there is a distribution $P$ for which the nearness constraints are met. The final two terms of (8) are normalisation constraints for $Q_i$ and $P$.

Taking derivatives with respect to $Q_i(y)$ and setting to zero gives

$$Q_i(y) = \frac{1}{Z_i} P(y)^{\frac{\rho_i}{1+\rho_i}} P_i(y)^{\frac{\lambda_i}{1+\rho_i}} \tag{9}$$

where $Z_i$ is a normalisation constant.

Given these $Q_i$, we can find also find an optimal, best fitting $P$. Taking derivatives of the Lagrangian with respect to $P(y)$ and setting to zero gives

$$P(y) = \sum_i \frac{\rho_i}{\sum_{i'} \rho_{i'}} Q_i(y) = \sum_i w_i \frac{(P_i(y)^{\lambda_i})^{\gamma_i} P(y)^{1-\gamma_i}}{Z_i} \tag{10}$$

where $w_i = \rho_i / \sum_i' \rho_{i'}$, and $\gamma_i = 1/(1+\rho_i)$, and $Z_i = \sum_{y'} (P_i(y')^{\lambda_i})^{\gamma_i} P(y')^{1-\gamma_i}$. Comparing this with (3) we see that this form of maximum entropy distribution is equivalent to the Rényi divergence aggregator of annealed forms of $P_i$. The maximum entropy parameters of the aggregator could be obtained by solving for the constraints or estimated using test data from $P(y)$. Empirically we find that, if all the $P_i$ are trained on the same data, or on data subject to sample-selection bias (rather than say an annealed form of the required distribution), then $\lambda_i \approx 1$.

Note that the parameter $\rho_i$ controls the level of penalty there is for a mismatch between the biased distributions $Q_i$ and the distribution $P$. If all the $\rho_i$ are zero for all $i$ then this penalty is removed and the $Q_i$ can bear little resemblance to the $P$ and hence to one another. In this setting (10) becomes a standard mixture and the aggregator is a linear opinion pool. If however $\rho_i$ tends to a large value for all $i$, then the distributions $Q_i$ are required to be much more similar. In this setting (10) becomes like a log opinion pool.

**Interim Summary** We have shown that the Rényi divergence aggregator is not an arbitary choice of aggregating distribution. Rather it is the maximum entropy aggregating distribution when the individual agent distributions are expected to be biased using a sample selection mechanism.

## 6 Implementation

Renyi divergence aggregators can be implemented with direct optimization, stochastic gradient methods, or using a variational optimization for the sum of weighted divergences, which is described here. The weighted Rényi Divergence objective given by Definition 2 can be lower bounded using

$$\sum_i w_i D(P_i, Q) \geq \sum_{i,y} \frac{w_i \gamma_i}{\gamma_i - 1} Q_i(y) \log \frac{[P_i(y)^{\gamma_i} Q(y)^{1-\gamma_i}]}{Q_i(y)} \tag{11}$$

where we have introduced variational distributions $Q_i$, and used Jensen's inequality. Note equality is obtained in (11) for $Q_i(y) \propto P_i(y)^{\gamma_i} Q(y)^{1-\gamma_i}$. Optimizing for $Q$ gives $P(y) = Q_{\text{opt}}(y) = \sum_i w_i^* Q_i(y)$ with $w_i^* = w_i \gamma_i^{-1} / \sum_i w_i \gamma_i^{-1}$.

This leads to an iterative variational algorithm that is guaranteed (using the same arguments as EM, and using the convexity of to optimize (11): iteratively set $Q_i(y) \propto P_i(y)^{\gamma_i} Q(y)^{1-\gamma_i}$, and then set $Q(y) \propto \sum_i w_i^* Q_i(y)$. The optimization of the parameters $w_i^*$ also naturally fits within this framework. $Q(y)$ is a simple mixture of $Q_i(y)$. Hence given $Q_i(y)$, the optimal $w_i^*$ are given by the optimal mixture model parameters. These can be determined using a standard inner Expectation Maximization loop. In practice, we get faster convergence if we use a single loop. First set $Q_i(y) \propto P_i(y)^{\gamma_i} Q(y)^{1-\gamma_i}$. Second compute $q_{in} = w_i^* Q_i(y_n) / \sum_i w_i^* Q_i(y_n)$. Third set $w_i^* = \sum_n q_{in} / \sum_{in} q_{in}$. Finally set $Q(y) \propto \sum_i w_i^* \gamma_i Q_i(y)$. This is repeated until convergence. All constants of proportionality are given by normalisation constraints. Note that where computing the optimal $Q$ may be computationally prohibitive, this process also gives rise to an approximate divergence minimization approach, where $Q_i$ is constrained to a tractable family while the optimizations for $Q_i$ are performed.

## 7 Experiments

To test the practical validity of the maximum entropy arguments, the following three tasks were implemented.

**Task 1: Aggregation on simulated data** We aim to test the variation of the aggregator performance as the bias of the agent datasets is gradually changed. This requires that the data does not dramatically change across tests of different biases. We tested this process using a number of bias generation procedures, all with the same implication in terms of results.

The details of the data generation and testing is given in Algorithm 1. We used $N_A = 10$, $K = 64$, $N_{Va} = 100$, $P^*$ was a discretized $N(32, 64/7)$, $f_i(y)$ $U([0, 1])$ to generate the artificial data that gave the results displayed here. Equivalent results were found for all (non-trivial) parameter choices we tried, as well as using completely different data generation procedures generating biased agent data.

**Task 2: Aggregation on chords from Bach chorales** This task aims to accurately predict distributions of chords from Bach chorales [2]. The Bach chorales data was split equally and randomly into training and test distributions. Then training data from half of the chorales was chosen to be shared across all the agents. After that each agent received additional training data from a random half of the remaining chorales. Each agent was trained using a mixture of Bernoulli's with a randomized number of mixture components between 5 and 100, and a random regularisation parameter between 0 and 1. 10 agents were used and after all 10 agents were fully trained, the Rényi mixture

**Algorithm 1** Generate test data for agents with different biases, and test aggregation methods.

Select a target discrete distribution $P^*(.)$ over $K$ values. Choose $N_A$, the number of agents.
Sample IID a small number $N_{Va}$ of values from the target distribution to get a validation set $\mathcal{D}_{Va}$
Sample IID a large number $N$ of values $\{y_n; n = 1, 2, 3, \ldots, N\}$ from the target distribution to get the base set $\mathcal{D}$ from which agent data is generated.
Sample bias probabilities $f_i(y)$ for each agent to be used as a rejection sampler.
**for** annealing parameter $\beta = 0$ TO 4 **do**
  **for** each agent $i$ **do**
    Anneal $f_i$ to get $f_k^*(y) = f_k(y)^\beta ./ \max_y f_i(y)^\beta$.
    For each data point $y_i$, reject it with probability $(1 - f_k^*(y_i))$.
    Collect the first 10000 unrejected points, and set $P_i$ to be the resulting empirical distribution.
    This defines the distribution $P_i$ for agent $i$ given the value of $\beta$.
  **end for**
  Find aggregate $P(.)$ for different aggregators given agent distributions $P_i$ and an additional $P_0$ corresponding to just the uniform distribution, using the validation dataset $\mathcal{D}_{Va}$ for any parameter estimation.
  Evaluate the performance of each aggregator using the KL Divergence between the target distribution $P^*(.)$ and the aggregate distribution $P(.)$: $KL(P^*||P)$.
**end for**

**Algorithm 2** Competition Data Preparation

Load image data. Discretize to 64 gray scales. Put in INT8 format. Define stopping criterion $\epsilon$
**for** j=1 to 140000 **do**
  Pick random image and random pixel at least 40 pixels away from edge of image and find $35 \times 30$ patch including that pixel at the bottom-middle of the patch.
  Record $x(j)$ =vectorisation of all pixels in patch 'before' that pixel in patch in raster-scan terms, $y(j)$ =grayscale value at chosen pixel,$i(j)$ =image number
**end for**
Produce three Matlab datasets. Set 1: $x$ and $y$ and $i$ values in one `.mat` for 100000 training records. Set 2: $x$ and $i$ values in one `.mat` file for 40000 test records. Set 3: $y$ values for the corresponding test cases, not publicly available.

weights were optimized using the whole training dataset. Performance results were computed on the held out test data.

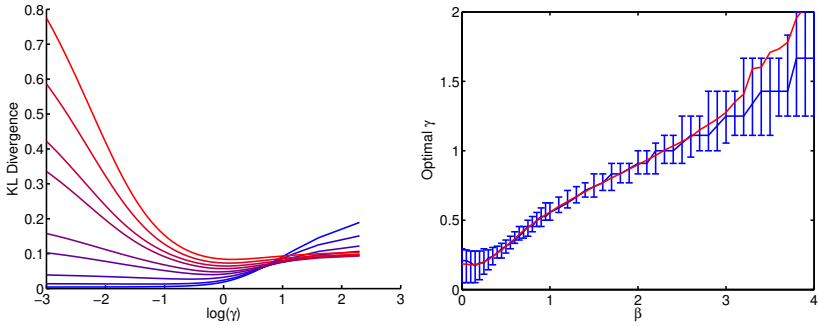
**Task 3: Aggregation on Kaggle competition** To analyze the use of combination methods in a realistic competition setting, we need data from an appropriate competitive setup. For this purpose we designed and ran the Kaggle-in-Class competition. The competition consisted of a critical problem in low-level image analysis: the image coding problem, which is fundamental in image compression, infilling, super-resolution and denoising. We used data consisting of images from van Hateren's Natural Image Dataset[2] [15]. The data was preprocessed using Algorithm 2 to put it in a form suitable for a Kaggle competition, and ensure the data sizes were sufficient for use on student machines, and that submission files were suitable for uploading (this is the reason for the 6 bit grayscale representation).

The problem was to provide a probabilistic prediction on the next pixel $y$ given information from previous pixels in a raster scan. The competitor's performance was measured by the perplexity on a public set at submission time, but the final ranked ordering was on a private test set. We chose as agent distributions the 269 submissions that had perplexity greater than that given by a uniform distribution and analysed the performance of a number of aggregation methods for the competition: weighted Rényi divergence aggregators, simple averaging of the top submissions (with an optimized choice of number), and a form of heuristic Bayesian model averaging, via an annealed likelihoood: $P(y|\cdot) \propto \sum_j P_j(y|\cdot) \left(P(j|\mathcal{D}_{\mathrm{tr}})\right)^{\alpha}$, where $\alpha$ is an aggregation parameter choice. The weighted Rényi divergence aggregators were optimized using stochastic gradient methods, until the change between epochs became negligible. The validation set ($20,000$ pixels) is used for learning the aggregation parameters. The test set (also $20,000$ pixels) is only used for the test results.

**Results** For Task 1, Figure 1(a) shows the test performance on different biases for different values of $\log(\gamma_i)$ in (10), where all $\gamma_i$ are taken to be identical and equal to $\gamma$. Figure 1(b) shows how the optimal value of $\gamma$ changes, as the bias parameter $\beta$ changes. Parameter optimization was done using a conjugate gradient method. The cost of optimization for Rényi mixtures is comparable to that of log opinion pools. For Task 2, Figure 2(a) shows the performance on the Bach chorales with 10 agents, with the implementation described in the Implementation section. Again in this real data setting, the Rényi mixtures show improved performance.

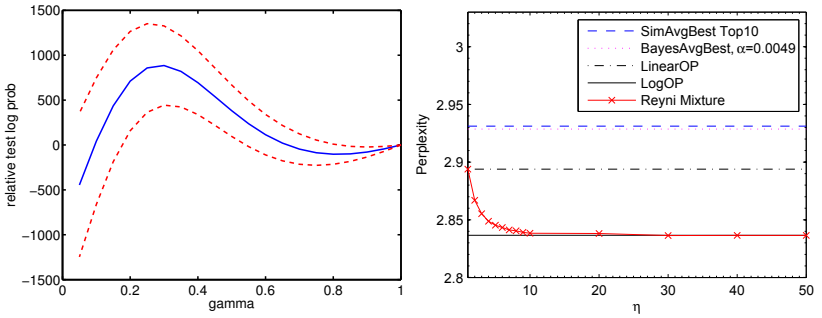
The two demonstrations show that when agents received a biased subsample of the overall data then Rényi-mixtures perform best as an aggregation method, in that they give the lowest KL divergence. As the bias increases, so the optimal value of $\gamma$ increases. In the limit that the agents see almost the same data from the target distribution, Rényi-mixtures with small $\gamma$ perform the best, and are

[2] `http://bethgelab.org/datasets/vanhateren/`




(a) (b)

**Fig. 1.** (a) Task 1: Plot of the KL divergence against $\log \gamma$ for one dataset with $\beta = 0$ (lower lines, blue) through to $\beta = 4$ (upper lines, red) in steps of 0.5. Note that, unsurprisingly, more bias reduces performance. However the optimal value of $\gamma$ (lowest KL), changes as $\beta$ changes. for low values of $\beta$ the performance of $\gamma = 0$ (log opinion pools) is barely distinguishable from other low $\gamma$ values. Note that using a log opinion pool (low $\gamma$) when there is bias produces a significant hit on performance. (b) Task 1: Plot of the optimal $\gamma$ (defining the form of Rényi mixture) for different values of $\beta$ (determining the bias in the generated datasets for each agent). The red (upper) line is the mean, the blue line the median and the upper and lower bars indicate the 75th and 25th percentiles, all over 100 different datasets. For $\beta = 0$ (no bias) we have optimal aggregation with lower $\gamma$ values, approximately corresponding to a log opinion pool. As $\beta$ increases, the optimal $\gamma$ gets larger, covering the full range of Rényi Mixtures.




(a) (b)

**Fig. 2.** (a) Task 2: test log probability results (relative to the log probability for a mixture) for the Bach chorales data for different values of $\gamma$, indicating the benefit of Rényi mixtures over linear ($\gamma = 1$) and log ($\gamma = 0$) opinion pools. Error bars are standard errors over 10 different allocations of chorales to agents prior to training. (b) Task 3: perplexity on the test set of all the compared aggregation methods against $\eta = 1/\gamma$. For each method, the best performance is plotted. Log opinion pools perform best as suggested by the maximum entropy arguments, and is statistically significantly better than the linear opinion pool($p = 8.0 \times 10^{-7}$). All methods perform better than the best individual competition entry (2.963).

indistinguishable from the $\gamma = 0$ limit. Rényi mixtures are equivalent to log opinion pools for $\gamma \to 0$.

For Task 3, all agents see unbiased data and so we would expect log opinion pools to be optimal. The perplexity values as a function of $\eta = 1/\gamma$ for all the methods tested on the test set can be seen in Figure 2(b). The parameter-based pooling methods perform better than simple averages and all forms of heuristic model averaging as these are inflexible methods. There is a significant performance benefit of using logarithmic opinion pooling over linear pooling, and weighted Rényi divergence aggregators interpolate between the two opinion pooling methods. This figure empirically supports the maximum entropy arguments.

## 8 Machine Learning Markets and Rényi Divergence Aggregation

Machine learning markets with isoelastic utilities [28] are an information market based aggregation method. Independent agents with different beliefs trade in a securities market. The equilibrium prices of the goods in that securities market can then be taken as an aggregate probability distribution, aggregating the individual agent beliefs. Following the notation and formalism in Storkey [28], agents indexed by $i$ with belief $P_i(y)$, wealth $W_i$ and utility function $U_i(.)$ trade in Arrow-Debreu securities derived from each possible outcome of an event. Given the agents maximize expected utility, the market equilibrium price of the securities $c(y)$ is used as an aggregate model $P(y) = c(y)$ of the agent beliefs. When each agent's utility is an isoelastic utility of the form $U_i(W) = W^{1-\eta_i}/(1-\eta_i)$ with a risk-averseness parameter $\eta_i$, the market equilibrium $P(y)$ is implicitly given by

$$P(y) = \sum_i \frac{W_i}{\sum_l W_l} \frac{P_i(y)^{\gamma_i} P(y)^{1-\gamma_i}}{\sum_{y'} P_i(y')^{\gamma_i} P(y')^{1-\gamma_i}} \tag{12}$$

with $\gamma_i = \eta_i^{-1}$ (generalising (10) in [28]). This shows the isoelastic market aggregator linearly mixes together components that are implicitly a weighted product of the agent belief and the final solution. Simple comparison of this market equilibrium with the Rényi Divergence aggregator (3) shows that the market solution and the Rényi divergence aggregator are of exactly the same form.

We conclude that a machine learning market implicitly computes a Rényi divergence aggregation via the actions of individual agents. The process of obtaining the market equilibrium is a process for building the Rényi Divergence aggregator, and hence machine learning markets provide a method of implementation of weighted Rényi divergence aggregators. The benefit of market mechanisms for machine learning is that they are *incentivized*. There is no assumption that the individual agents behave cooperatively, or that there is an overall controller who determines agents' actions. Simply, if agents choose to maximize their utility (under myopic assumptions) then the result is weighted Rényi Divergence aggregation.

In general, equilibrium prices are not necessarily straightforward to compute, but the algorithm in the implementation section provides one such method. As this iterates computing an interim $P$ (corresponding to a market price) and an interim $Q_i$ corresponding to agent positions given that price, the mechanism in this paper can lead to a form of tâtonnement algorithm with a guaranteed market equilibrium – see e.g. [6].

The direct relationship between the risk averseness parameter for the isoelastic utilities and the bias controlling parameter of the Rényi mixtures ($\gamma_i = \eta_i^{-1}$) provides an interpretation of the isoelastic utility parameter: if agents know they are reasoning with respect to a biased belief, then an isoelastic utility is warranted, with a choice of risk averseness that is dependent on the bias.

In [28] the authors show, on a basket of UCI datasets, that market aggregation with agents having isoelastic utilities performs better than simple linear opinion pools (markets with log utilities) and products (markets with exponential utilities) when the data agents see is biased. As such markets implement Rényi mixtures, this provides additional evidence that Rényi mixtures are appropriate when combining biased predictors.

## 9 Discussion

When agents are training and optimising on different datasets than one another, log opinion pooling is no longer a maximum entropy aggregator. Instead, under certain assumptions, the weighted Rényi divergence aggregator is the maximum entropy solution, and tests confirm this practically. The weighted Rényi divergence aggregator can be implemented using isoelastic machine learning markets.

Though there is some power in providing aggregated prediction mechanisms as part of competition environments, there is the additional question of the competition mechanism itself. With the possibility of using the market-based aggregation mechanisms, it would be possible to run competitions as prediction market or collaborative scenarios [1], instead of as winner takes all competitions. This alternative changes the social dynamics of the system and the player incentives, and so it is an open problem as to the benefits of this. We recognize the importance of such an analysis as an interesting direction for future work.

## References

1. Abernethy, J., Frongillo, R.: A collaborative mechanism for crowdsourcing prediction problems. In: Advances in Neural Information Processing Systems 24 (NIPS2011) (2011)
2. Bache, K., Lichman, M.: UCI machine learning repository (2013), `http://archive.ics.uci.edu/ml`
3. Barbu, A., Lay, N.: An introduction to artificial prediction markets for classification (2011), arXiv:1102.1465v3
4. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)

5. Chen, Y., Wortman Vaughan, J.: A new understanding of prediction markets via no-regret learning. In: Proceedings of the 11th ACM conference on Electronic commerce (2010)
6. Cole, R., Fleischer, L.: Fast-converging tatonnement algorithms for the market problem. Tech. rep., Dept. Computer Science. Dartmouth College. (2007)
7. Dani, V., Madani, O., Pennock, D., Sanghai, S., Galebach, B.: An empirical comparison of algorithms for aggregating expert predictions. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI) (2006)
8. Dietrich, F.: Bayesian group belief. Social choice and welfare 35(4), 595–626 (2010)
9. Dietterich, T.: Ensemble methods in machine learning. In: Lecture Notes in Computer Science, vol. 1857, pp. 1–5. Springer Verlag (2000)
10. Domingos, P.: Why does bagging work? a Bayesian account and its implications. In: Proceedings KDD (1997)
11. Everingham, M., et al.: The 2005 PASCAL visual object classes challenge. In: Selected Proceedings of the first PASCAL Challenges Workshop LNAI. pp. 117–176. No. 3944 (2006)
12. Garg, A., Jayram, T., Vaithyanathan, S., Zhu, H.: Generalized opinion pooling. AMAI (2004)
13. Goldbloom, A.: Data prediction competitions – far more than just a bit of fun. In: IEEE International Conference on Data Mining Workshops (2010)
14. Green, K.: The $1 million Netflix challenge. Technology Review (2006)
15. Hateren, J.H.v., Schaaf, A.v.d.: Independent component filters of natural images compared with simple cells in primary visual cortex. Proceedings: Biological Sciences 265(1394), 359–366 (Mar 1998)
16. Heskes, T.: Selecting weighting factors in logarithmic opinion pools. In: Advances in Neural Information Processing Systems 10 (1998)
17. Kahn, J.M.: A generative Bayesian model for aggregating experts' probabilities. In: Proceedings of the 20th conference on Uncertainty in Artificial Intelligence. pp. 301–308. AUAI Press (2004)
18. Lay, N., Barbu, A.: Supervised aggregation of classifiers using artificial prediction markets. In: Proceedings of ICML (2010)
19. Maynard-Reid, P., Chajewska, U.: Aggregating learned probabilistic beliefs. In: Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence. pp. 354–361. Morgan Kaufmann Publishers Inc. (2001)
20. Ottaviani, M., Sørensen, P.: Aggregation of information and beliefs in prediction markets (2007), fRU Working Papers
21. Pennock, D., Wellman, M.: Representing aggregate belief through the competitive equilibrium of a securities market. In: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence. pp. 392–400 (1997)
22. Robert M. Bell, Y.K., Volinsky, C.: All together now: A perspective on the NETFLIX PRIZE. Chance 24 (2010)
23. Rubinstein, M.: An aggregation theorem for securities markets. Journal of Financial Economics 1(3), 225–244 (1974)
24. Rubinstein, M.: Securities market efficiency in an Arrow-Debreu economy. American Economic Review 65(5), 812–824 (1975)
25. Rubinstein, M.: The strong case for the generalised logarithmic utility model as the premier model of financial markets. Journal of Finance 31(2), 551–571 (1976)
26. Storkey, A.: When training and test sets are different: Characterising learning transfer. In: Lawrence, C.S.S. (ed.) Dataset Shift in Machine Learning, chap. 1, pp. 3–28. MIT Press (2009), http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=11755

27. Storkey, A.: Machine learning markets. In: Proceedings of Artificial Intelligence and Statistics. vol. 15. Journal of Machine Learning Research W&CP (2011), http://jmlr.csail.mit.edu/proceedings/papers/v15/storkey11a/storkey11a.pdf
28. Storkey, A., Millin, J., Geras, K.: Isoelastic agents and wealth updates in machine learning markets. In: Proceedings of ICML 2012 (2012)
29. West, M.: Bayesian aggregation. Journal of the Royal Statistical Society 147, 600–607 (1984)
30. Wolpert, D.H.: Stacked generalization. Neural Networks 5(2), 241 – 259 (1992), http://www.sciencedirect.com/science/article/pii/S0893608005800231